BERT 模型的数学形式

何沧平
cangping@staff.weibo.com
微博

许涛
xutao@sugon.com
曙光信息产业（北京）有限公司

摘　要

最近流行的自然语言处理技术之一是 BERT 模型，本文给出该模型的数学形式。

关键词： BERT、自然语言处理

Mathematical Principles of BERT Model*

He Cangping
cangping@staff.weibo.com
WEIBO.COM

Xu Tao
xutao@sugon.com
SUGON.COM

Abstract

BERT is the most popular natural language processing(NLP) model In the recent 3 years. This paper presents its mathematic formulas in detail.

Keywords: BERT, natural language processing

1 引言

在自然语言处理领域，BERT[1] 模型是最近两三年的流行技术。它为后续一大批模型带来灵感，例如 ALBERT[2]、XLNET[3]、RoBERTa[4]。

BERT 原论文没有详细描述模型细节。BERT 模型的主体来自于自注意力编码器 [5]，然而论文 [5] 也是用自然语言大致描述，没有给出具体细节。BERT 作者提供的 TensorFlow 代码[1]中的实现方式，与原论文 [1] 中的描述也有差异。

为了迅速应用于业务、严谨地理论研究，本文给出 BERT 模型的数学形式，将程序代码改写为数字公式。程序代码与原论文不一致的地方，以程序代码为准。

*完稿日期：2020 年 12 月 16 日

[1] https://github.com/google-research/bert

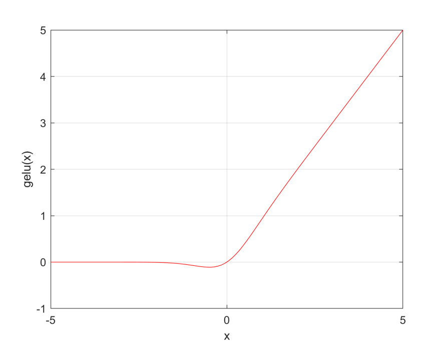


图 1: 激活函数 gelu

2 函数定义

作为准备，本节定义几个函数。目前 BERT 代码中数组的组织方式是行优先，因此本文中的向量、矩阵也按行优先来定义。

任意给定正整数 m 和 n，行向量用黑体小写字母表示，形式为 $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$。矩阵用大写字母表示，形式为

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}.$$

软大函数 (softmax) 定义为

$$\begin{aligned} \operatorname{smax}(\boldsymbol{x}) &= \frac{1}{\sum_{i=1}^{i=n} e^{x_i}} (e^{x_1}, e^{x_2}, \ldots, e^{x_n}), \\ \operatorname{smax}(X) &= \begin{bmatrix} \operatorname{smax}(x_{1:}) \\ \operatorname{smax}(x_{2:}) \\ \vdots \\ \operatorname{smax}(x_{m:}) \end{bmatrix} = (\operatorname{smax}(x_{1:}); \operatorname{smax}(x_{2:}); \ldots; \operatorname{smax}(x_{m:})), \end{aligned}$$

这里的 $x_{i:} = (x_{i1}, x_{i2}, \ldots, x_{in})$，圆括号里的分号表示换行。

对向量或矩阵求对数时，对数作用到它们的每一个元素上，即

$$\begin{aligned} \log(\boldsymbol{x}) &= (\log(x_1), \log(x_2), \ldots, \log(x_n)), \\ \log(X) &= \begin{bmatrix} \log(x_{11}) & \log(x_{12}) & \cdots & \log(x_{1n}) \\ \log(x_{21}) & \log(x_{22}) & \cdots & \log(x_{2n}) \\ \vdots & \vdots & & \vdots \\ \log(x_{m1}) & \log(x_{m2}) & \cdots & \log(x_{mn}) \end{bmatrix}. \end{aligned}$$

对实数 x, 激活函数

$$\operatorname{gelu}(x) = 0.5x(1 + \tanh[\sqrt{0.5\pi}(x + 0.044715x^3)]),$$

gelu 的图像见图1.

函数 gelu 作用到的向量和矩阵上时，它作用到每一个元素上。

层归一化 (layer normalization) 函数

$$\mathrm{lnor}(X)=\begin{bmatrix} \frac{\gamma_1(x_{11}-\mu_1)}{\sigma_1}+\beta_1 & \frac{\gamma_2(x_{12}-\mu_2)}{\sigma_1}+\beta_2 & \cdots & \frac{\gamma_n(x_{1n}-\mu_n)}{\sigma_n}+\beta_n \\ \frac{\gamma_1(x_{21}-\mu_1)}{\sigma_1}+\beta_1 & \frac{\gamma_2(x_{22}-\mu_2)}{\sigma_1}+\beta_2 & \cdots & \frac{\gamma_n(x_{2n}-\mu_n)}{\sigma_n}+\beta_n \\ \vdots & \vdots & & \vdots \\ \frac{\gamma_1(x_{m1}-\mu_1)}{\sigma_1}+\beta_1 & \frac{\gamma_2(x_{m2}-\mu_2)}{\sigma_1}+\beta_2 & \cdots & \frac{\gamma_n(x_{mn}-\mu_n)}{\sigma_n}+\beta_n \end{bmatrix},$$

这里的 $\mu_j=\frac{1}{m}\sum_{i=1}^{m}x_{ij},\sigma_j=\sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_{ij}-\mu_j)^2},j=1,2,\ldots,n$。$\boldsymbol{\beta}=(\beta_1,\beta_2,\ldots,\beta_n)$ 和 $\boldsymbol{\gamma}=(\gamma_1,\gamma_2,\ldots,\gamma_n)$ 是待定向量。

对任意实数 x 和任意实数 $\alpha\in[0,1)$，随机取舍 (dropout) 函数定义为

$$\mathrm{drp}(x,\alpha)=\begin{cases}0, & \text{以概率 } \alpha \text{ 取此值},\\ \frac{x}{1-\alpha}, & \text{以概率 } 1-\alpha \text{ 取此值}.\end{cases}$$

drp 简称随取函数。对任意矩阵 X 和任意实数 $\alpha\in[0,1)$，随取函数作用到每个元素上

$$\mathrm{drp}(X,\alpha)=\begin{bmatrix}\mathrm{drp}(x_{11},\alpha) & \mathrm{drp}(x_{12},\alpha) & \cdots & \mathrm{drp}(x_{1n},\alpha)\\ \mathrm{drp}(x_{21},\alpha) & \mathrm{drp}(x_{22},\alpha) & \cdots & \mathrm{drp}(x_{2n},\alpha)\\ \vdots & \vdots & & \vdots \\ \mathrm{drp}(x_{m1},\alpha) & \mathrm{drp}(x_{m2},\alpha) & \cdots & \mathrm{drp}(x_{mn},\alpha)\end{bmatrix}.$$

假设行向量 $\hat{\boldsymbol{x}}=(\hat{x}_1,\hat{x}_2,\ldots,\hat{x}_n)$，将行向量与矩阵相加定义为逐行相加，即

$$X+\hat{\boldsymbol{x}}=\begin{bmatrix}x_{11}+\hat{x}_1 & x_{12}+\hat{x}_2 & \cdots & x_{1n}+\hat{x}_n\\ x_{21}+\hat{x}_1 & x_{22}+\hat{x}_2 & \cdots & x_{2n}+\hat{x}_n\\ \vdots & \vdots & & \vdots\\ x_{m1}+\hat{x}_1 & x_{m2}+\hat{x}_2 & \cdots & x_{mn}+\hat{x}_n\end{bmatrix}.$$

3 BERT 模型全貌

定义几个常数，并给出典型值。典型值是 google 预训练模型的一种参数配置，其它参数配置见 BERT 源码网站。n_1 为词碎数量，典型值 30522；n_2 为词碎嵌入宽度，典型值 512；n_3 为词碎序列长度，典型值 128；n_4 为被遮挡语碎数量，典型值 20；n_5 为自注意力头数，典型值 8；n_6 为单头宽度，等于 $\frac{n_2}{n_5}$，典型值 64；n_7 为全连接层宽数，典型值 2048；n_8 为编码器层数，典型值 8。

BERT 模型的全貌见图2。BERT 模型的输入是词碎序列，形式为

[CLS]␣ 词碎 2␣ 词碎 3 ⋯ 词碎 63␣[SEP] 词碎 65␣ 词碎 66⋯ 词碎 127␣[SEP]

␣是显式空格，用来分隔词碎。第 1 个 [SEP]（含）之前的词碎序列称为上句，第 1 个 [SEP]（不含）之后的词碎序列称为下句。中间 [SEP] 的位置只是示意，不要求一定是第 64 个词碎。词碎序列中，一些随机位置上是 [MASK]，它表示原来句中的词碎被“遮挡”了。例如：

[CLS]␣[MASK]␣[MASK]␣济␣南␣的␣雪␣[MASK]␣下␣在␣了␣这␣里␣，␣跑␣马␣岭␣位␣于␣济␣南␣南␣部␣山␣区␣[MASK]␣海␣拔␣近␣[MASK]␣米␣。␣[SEP]␣[MASK]␣素␣裹␣的␣[MASK]␣[MASK]␣云␣雾␣飘␣渺␣，␣山␣中␣民␣[MASK]␣木␣屋␣格␣外␣精␣致␣，␣不␣用␣[MASK]␣了␣[MASK]␣计␣去␣砍␣树␣赚␣钱␣，

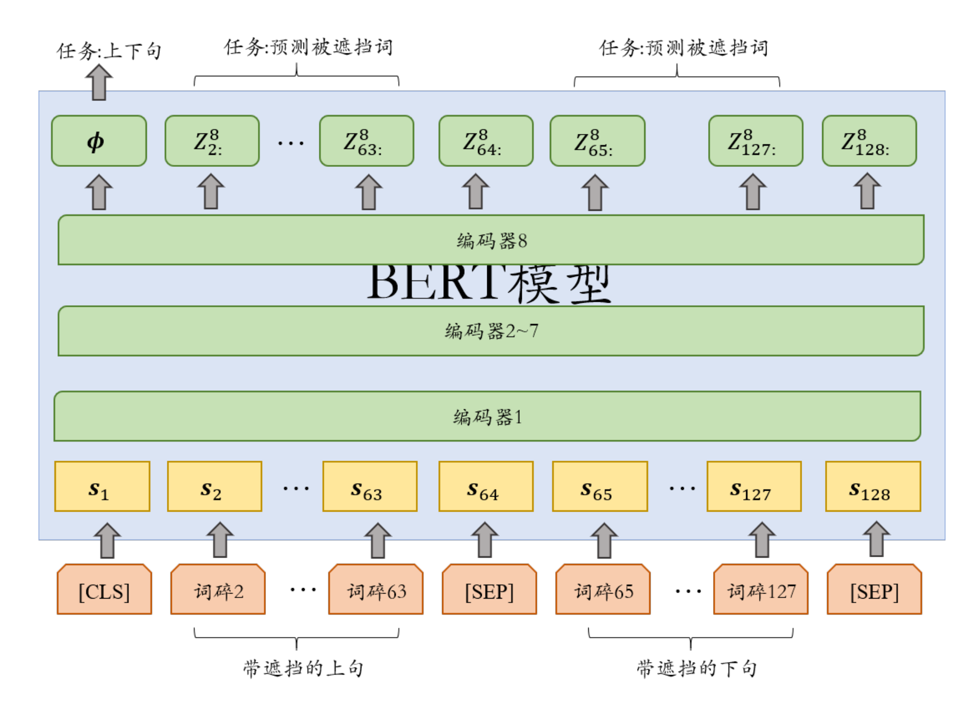


图 2: BERT 模型全貌。词碎序列长度为 128，编码器层数为 8，这 2 个数字均可以按需调整。

␣更␣不␣用␣与␣熊␣大␣熊␣二␣斗␣[MASK]␣斗␣勇␣费␣心␣劳␣[MASK]␣,␣在␣这␣可␣[MASK]␣像␣光␣头␣强␣一␣样␣惬␣意␣的␣蜗␣居␣一␣天␣,␣享␣受␣[MASK]␣段␣美␣好␣的␣冰␣[MASK]␣奇␣缘␣[MASK]␣#␣济␣南␣爆␣[MASK]␣#␣#␣忍␣不␣住␣想␣拍␣的␣[SEP]

这个例子中，上句是

[CLS]␣[MASK]␣[MASK]␣济␣南␣的␣雪␣[MASK]␣下␣在␣了␣这␣里␣,␣跑␣马␣岭␣位␣于␣济␣南␣南␣部␣山␣区␣[MASK]␣海␣拔␣近␣[MASK]␣米␣。␣[SEP]

下句是

␣[MASK]␣素␣裹␣的␣[MASK]␣[MASK]␣云␣雾␣飘␣渺␣,␣山␣中␣民␣[MASK]␣木␣屋␣格␣外␣精␣致␣,␣不␣用␣[MASK]␣了␣[MASK]␣计␣去␣砍␣树␣赚␣钱␣,␣更␣不␣用␣与␣熊␣大␣熊␣二␣斗␣[MASK]␣斗␣勇␣费␣心␣劳␣[MASK]␣,␣在␣这␣可␣[MASK]␣像␣光␣头␣强␣一␣样␣惬␣意␣的␣蜗␣居␣一␣天␣,␣享␣受␣[MASK]␣段␣美␣好␣的␣冰␣[MASK]␣奇␣缘␣[MASK]␣#␣济␣南␣爆␣[MASK]␣#␣#␣忍␣不␣住␣想␣拍␣的␣[SEP]

上句和下句来自紧下方这条微博中的 3 句话，并将长度裁剪为 128.

原来济南的雪都下在了这里，跑马岭位于济南南部山区，海拔近千米。

有幸亲临今冬初雪，银装素裹的山林云雾飘渺，山中民宿木屋格外精致，不用为了生计去砍树赚钱，更不用与熊大熊二斗智斗勇费心劳神，在这可以像光头强一样惬意的蜗居一天，享受这段美好的冰雪奇缘。

济南爆料 ## 忍不住想拍的冬日美景 #@ 济南文旅发展集团

词碎进入 BERT 之后，立即被转化为向量。相应地，词碎序列转化为矩阵 S，S 的每一个行向量对应一个词碎。接下来，矩阵 S 被喂给第 1 个编码器，第 1 个编码器输出矩阵 Z^1，矩阵 Z^1 随后被喂给第 2 个编码器。这样依次操作，第 n_8 个编码器的输出为 Z^{n_8}，这也是 BERT 模型的输出。图2中的向量 $\boldsymbol{\phi}$ 是 Z^{n_8} 的第 1 行。

为了得到 BERT 模型内部参数的最优值，需要 2 个任务来设定优化目标函数。上下句任务：判断词碎序列中的上句和下句是否为真实的承接关系。这要求制作一批正样本和一批负样本，正样本从相邻的句子中产生，负样从不同的文档中产生。遮挡任务：原论文称为“遮挡的语言模型”，猜测被遮挡的词。

4 制作训练样本

训练语料通常来自多篇文章，每一篇文章都分割为多个句子。例如每条微博文本都可以视为一篇文章，话题、句号、问号均可以作为句子结束的标志。

从训练语料中生成一个词碎词典 $\mathcal{C} = \{c_1, c_2, \ldots, c_{n_1}\}$，具体的生成方法有字对编码 [6][7]（Byte Pair Encoding）、WordPiece 和 Unigram Language Model。词典中还要包含几个特殊的词碎：[CLS]、[SEP]、[MASK]、[PAD]、[UNK]。对中文来说，词碎是单个字、单个标点符号。对英文来说，词碎是组成单词的片段，例如 un、##aff、##able，2 个井号表示该词碎应该接在别的词碎后面。任何一个单词都可以分割成若干词碎，例如 unaffable 能分割成 un␣##aff␣##able。

将训练语料的中文句子和英文句子全部转化为词碎句子，此后提及的训练语料均指词碎形式的训练练语料。

词典的中每个词碎 c_i 都嵌入到一个行向量 $\boldsymbol{d}_i$，$\boldsymbol{d}_i$ 的尺寸为 $1 \times n_2$，尺寸典型值为 1×512。将所有的行向量 $\boldsymbol{d}_i$ 按顺序排列起来，组成矩阵 $D = (\boldsymbol{d}_1; \boldsymbol{d}_2; \ldots; \boldsymbol{d}_{n_1})$，尺寸为 $n_1 \times n_2$，尺寸典型值 30522×512。

训练样本分正样本和负样本，正样本词碎序列的上句和下句是真实的承接关系，负样本词碎序列的上句和下句之间没有承接关系。

从训练语料的同一篇文章中挑出 2 个相邻的句子适当裁剪，使 2 个句子的长度等于 $n_3 - 3$，典型值为 125。按照形式“[CLS]␣ 第 1 个句子 ␣[SEP]␣ 第 2 个句子 ␣[SEP]”组成序列 $\boldsymbol{\tau}$，记为

$$\boldsymbol{\tau} = \tau_1 \tau_2 \ldots \tau_{n_3},$$

这里的 $\tau_i \in \mathcal{C}, i = 1, 2, \ldots, n_3$. 从 $\boldsymbol{\tau}$ 随机挑出 n_4 个词碎，要求不能是 [CLS]、[SEP]、[PAD]。这 n_4 个词碎在 $\boldsymbol{\tau}$ 中的位置编号记为 $\hat{\boldsymbol{t}} = (\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_{n_4})$，在词典 $\mathcal{C}$ 中的编号记为 $\tilde{\boldsymbol{t}} = (\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_{n_4})$，显然 $\hat{t}_i \in \{1, 2, \ldots, n_3\}$，$\tilde{t}_i \in \{1, 2, \ldots, n_1\}$。

对这 n_4 个位置 $(\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_{n_4})$，随机挑出 $0.8 \times n_4$ 个并将 $\boldsymbol{\tau}$ 中的对应词碎替换为 [MASK]，随机挑出 $0.1 \times n_4$ 个并将 $\boldsymbol{\tau}$ 中的对应词碎并替换为 $\mathcal{C}$ 中的其它词碎；剩余 $0.1 \times n_4$ 个位置，$\boldsymbol{\tau}$ 中的对应词碎保持不变。将替换后的词碎序列记为 $\hat{\boldsymbol{\tau}}$，即得到一个正样本。

如果从训练语料的不同文章中挑出 2 个句子，并用同样的方法制作成词碎序列 $\hat{\boldsymbol{\tau}}$，就得到一个负样本。

第 3 节的语碎序列中，随机挑出来的遮挡的位置为

$$\hat{\boldsymbol{t}} = (2, 3, 8, 26, 30, 34, 38, 39, 48, 57, 60, 77, 83, 88, 105, 111, 114, 116, 119),$$

对应的词碎为

原␣来␣都␣，␣千␣装␣山␣林␣宿␣为␣生␣智␣神␣以␣这␣雪␣。␣济␣料

这些词碎在词典 $\mathcal{C}$ 中的编号 $\tilde{\boldsymbol{t}} = (1334, 3342, 6964, 8025, 1284, 6164, 2256, 3361, 2163, 712, 4496, 3256, 4869, 810, 6822, 7435, 512, 3846, 3161)$。注意，词碎“济”保持不变。

5 编码器

输入 BERT 模型的样本是词碎序列，不能直接进行矩阵、向量运算，需要先转换成矩阵形式。这个转换工作在第 1 个编码器前完成。

5.1 输入向量

给定训练样本 $\hat{\boldsymbol{\tau}} = \hat{\tau}_1\hat{\tau}_2\ldots\hat{\tau}_{n_3}$。对 $i = 1, 2, \ldots, n_3$，取出 $\hat{\tau}_i$ 在矩阵 D 中的对应行向量，记为 $\boldsymbol{s}_i$。将 $\boldsymbol{s}_i$ 按 i 从小到大顺序排列起来，得到矩阵 $S = (\boldsymbol{s}_1; \boldsymbol{s}_2; \ldots; \boldsymbol{s}_{n_3})$，$S$ 的尺寸为 $n_3 \times n_2$，尺寸典型值 128×512。

位置矩阵记为 P，尺寸为 $n_3 \times n_2$，尺寸典型值 128×512。P 的每一行对应词碎序列中的一个位置。称行向量 $\boldsymbol{f}_1$ 和 $\boldsymbol{f}_2$ 为句标向量，向量尺寸为 $1 \times n_3$，尺寸典型值 1×512。$\boldsymbol{f}_1$ 对应词碎序列中的上句，$\boldsymbol{f}_2$ 对应词碎序列中的下句。记 $F = (\boldsymbol{f}_1; \boldsymbol{f}_1; \ldots; \boldsymbol{f}_1; \boldsymbol{f}_2; \ldots; \boldsymbol{f}_2)$，尺寸为 $n_3 \times n_2$，尺寸典型值 128×512。对任意 $i = 1, 2, \ldots, n_3$，如果 $\hat{\tau}_i$ 属于上句，那么 F 的第 i 行等于 $\boldsymbol{f}_1$; 如果 $\hat{\tau}_i$ 属于下句，那么 F 的第 i 行等于 $\boldsymbol{f}_2$。

令 $Z^0 = S + P + F$，尺寸为 $n_3 \times n_2$，尺寸典型值 128×512。Z^0 是第 1 个编码器的输入矩阵。自注意力层的随取概率记为 $\alpha_1 \in [0, 1)$, 全连接层的随取概率记为 $\alpha_2 \in [0, 1)$。在训练阶段，α_1 和 α_2 取值非零，官方代码中取值均为 0.1；在预测阶段，α_1 和 α_2 均取值为 0.

接下来的自注意力子层和全连接子层均指第 1 个编码器，不再每次说明。

5.2 自注意力子层

对 $i = 1, 2, \ldots, n_5$，第 i 头的“查”权重矩阵记为 W^{1i1}，尺寸 $n_2 \times n_6$，尺寸典型值 512×64；第 i 头的“查”偏置向量记为 $\boldsymbol{b}^{1i1}$，尺寸 $1 \times n_6$，尺寸典型值 1×64。“查”矩阵

$$Q^{1i} = Z^0 W^{1i1} + \boldsymbol{b}^{1i1},$$

尺寸 $n_3 \times n_6$，尺寸典型值 128×64.

对 $i = 1, 2, \ldots, n_5$，第 i 头的“键”权重矩阵记为 W^{1i2}，尺寸 $n_2 \times n_6$，尺寸典型值 512×64；第 i 头的“键”偏置向量记为 $\boldsymbol{b}^{1i2}$，尺寸 $1 \times n_6$，尺寸典型值 1×64。“键”矩阵

$$K^{1i} = Z^0 W^{1i2} + \boldsymbol{b}^{1i2},$$

尺寸 $n_3 \times n_6$，尺寸典型值 128×64.

对 $i = 1, 2, \ldots, n_5$，第 i 头的“值”权重矩阵记为 W^{1i3}，尺寸 $n_2 \times n_6$，尺寸典型值 512×64；第 i 头的“值”偏置向量记为 $\boldsymbol{b}^{1i3}$，尺寸 $1 \times n_6$，尺寸典型值 1×64。“值”矩阵

$$V^{1i} = Z^0 W^{1i3} + \boldsymbol{b}^{1i3},$$

尺寸 $n_3 \times n_6$，尺寸典型值 128×64.

记

$$R^{1i} = \mathrm{drp}\left(\mathrm{smax}\left(\frac{Q^{1i}(K^{1i})^T}{\sqrt{n_6}}\right), \alpha_1\right).$$

第 i 头的归一化分值为

$$U^{1i} = R^{1i}V^{1i},$$

尺寸为 $n_3 \times n_6$，尺寸典型值 128×64。将所有头的归一化分值连接起来，就得到第 1 个编码器自注意力的分值

$$U^1 = (U^{11}, U^{11}, \ldots, U^{1n_5}),$$

尺寸 $n_3 \times n_2$，尺寸典型值 128×512.

$W^{1\cdot 4}$ 为权重矩阵，尺寸 $n_2 \times n_2$，尺寸典型值 512×512。$\boldsymbol{b}^{1\cdot 4}$ 为第 1 个编码器偏置向量，尺寸 $1 \times n_2$，尺寸典型值 1×512。线性变换后施加随取操作，得

$$Y^{11} = \mathrm{drp}(U^1 W^{1\cdot 4} + \boldsymbol{b}^{1\cdot 4}, \alpha_2).$$

做一个层归一化操作，得到自注意力子层的输出

$$Y^{12} = \mathrm{lnor}(Y^{11} + Z^0),$$

尺寸 $n_3 \times n_2$，尺寸典型值 128×512.

5.3 全连接子层

$W^{1\cdot 5}$ 为全连接权重矩阵，尺寸 $n_2 \times n_7$，尺寸典型值 512×2048。$\boldsymbol{b}^{1\cdot 5}$ 为全连接偏置向量，尺寸 $1 \times n_7$，尺寸典型值 1×2048。全连接层的输出为

$$Y^{13} = \mathrm{gelu}(Y^{12} W^{1\cdot 5} + b^{1\cdot 5}),$$

尺寸 $n_3 \times n_7$，典型尺寸 128×2048.

$W^{1\cdot 6}$ 为线性变换权重矩阵，尺寸 $n_7 \times n_2$，尺寸典型值 2048×512。$\boldsymbol{b}^{1\cdot 6}$ 为线性变换偏置向量，尺寸 $1 \times n_2$，尺寸典型值 1×512。用线性变换将尺寸 $n_3 \times n_7$ 变回 $n_3 \times n_2$，然后作一下随取操作，即

$$Y^{14} = \mathrm{drp}(Y^{13} W^{1\cdot 6} + \boldsymbol{b}^{1\cdot 6}, \alpha_2).$$

施加一个层归一化操作，得到第 1 个编码器的输出

$$Z^1 = \mathrm{lnor}(Y^{14} + Y^{12}),$$

尺寸 $n_3 \times n_2$，尺寸典型值 128×512.

5.4 编码器堆叠

第 1 个编码器的输入是矩阵是 Z^0，输出矩阵是 Z^1。每个编码器内部的计算过程都一样，第 2 个编码器的输入矩阵是 Z^1，输出矩阵是 Z^2。依次类推，第 n_8 个编码器的输入矩阵是 Z^{n_8-1}，输出矩阵是 Z^{n_8}. 对 $j = 1, 2, \ldots, n_8$，矩阵 Z^j 的尺寸是 $n_3 \times n_2$，尺寸典型值是 128×512。

6 训练任务

训练 BERT 模型时，使用 2 个任务：上下句匹配、词碎遮挡。

6.1 上下句匹配任务

记 $\boldsymbol{\phi} = Z_{1:}^{n_8}$，即词碎 [CLS] 对应的向量。$\boldsymbol{\phi}$ 的尺寸为 $1 \times n_2$，尺寸典型值 1×512. 令 E 是尺寸为 $2 \times n_2$ 的矩阵，尺寸典型值为 2×512。令 η 是尺寸为 1×2 的向量。当输入序列 $\hat{\boldsymbol{\tau}}$ 的上句和下句是真实的承接关系时，$\boldsymbol{\xi} = (0, 1)$，否则 $\boldsymbol{\xi} = (1, 0)$。任务权重矩阵记为 $\hat{W}^1$，尺寸为 $n_2 \times n_2$，尺寸典型值 512×512。任务偏置向量记为 $\hat{\boldsymbol{b}}^1$，尺寸为 $1 \times n_2$，尺寸典型值 1×512。上下句匹配任务的目标函数为

$$l_1 = -\log(\mathrm{smax}(\tanh{(\boldsymbol{\phi}\hat{W}^1 + \hat{\boldsymbol{b}}^1)}E^T + \eta))\boldsymbol{\xi}^T.$$

6.2 词碎遮挡任务

从 Z^{n_8} 中取出行号为 $\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_{n_4}$ 的行向量，按顺序排列起来，组成矩阵 $\hat{Z}^2$，尺寸为 $n_4 \times n_2$，尺寸典型值 20×512。任务权重矩阵记为 $\hat{W}^2$，尺寸为 $n_2 \times n_2$，尺寸典型值 512×512；任务偏置向量记为 $\hat{\boldsymbol{b}}^2$，尺寸为 $1 \times n_2$，尺寸典型值 1×512. 词典 $\mathcal{C}$ 的偏置向量记为 $\boldsymbol{b}$，尺寸为 $1 \times n_1$，尺寸典型值 1×30522. 令

$$H = -\log(\mathrm{smax}(\mathrm{gelu}(\hat{Z}^2\hat{W}^2 + \hat{\boldsymbol{b}}^2)D^T + \boldsymbol{b})),$$

H 的尺寸为 $n_4 \times n_1$，尺寸典型值 20×30522.

根据编号向量 $\tilde{\boldsymbol{t}} = (\tilde{t}_1, \tilde{t}_2, \ldots, \tilde{t}_{n_4})$，从 H 中取出元素求平均值，就得本任务的目标函数，即

$$l_2 = \frac{1}{n_4}\sum_{i=1}^{n_4} h_{i,\tilde{t}_i},$$

这里的 $h_{i,\tilde{t}_i}$ 是矩阵 H 的元素。

6.3 模型预训练

以 $l_1 + l_2$ 为目标函数，以 $\hat{\boldsymbol{\tau}}$ 为样本进行训练，即可得到所有的待定参数。

7 待定参数的数量

第4节中，矩阵 D 参数量为 $n_1 n_2$。第5.1节中，矩阵 P，参数量为 $n_3 n_2$；句标向量 $\boldsymbol{f}_1$ 和 $\boldsymbol{f}_2$ 的参数量为 $2n_2$。

第5.2节中，矩阵 W^{1i1}、W^{1i2} 和 W^{1i3} 的参数量均为 $n_2 n_6$，向量 $\boldsymbol{b}^{1i1}$、$\boldsymbol{b}^{1i2}$ 和 $\boldsymbol{b}^{1i3}$ 的参数量均为 n_6，$W^{1\cdot4}$ 的参数量为 n_2^2，向量 $\boldsymbol{b}^{1\cdot4}$ 的参数量为 n_2；Y^{12} 对应的函数 lnor 中隐含 $2n_2$ 个参数。第5.3节中，矩阵 $W^{1\cdot5}$ 的参数量为 $n_2 n_7$，向量 $\boldsymbol{b}^{1\cdot5}$ 的参数量为 n_7; 矩阵 $W^{1\cdot6}$ 的参数量为 $n_7 n_2$，向量 $\boldsymbol{b}^{1\cdot6}$ 的参数量为 n_2；Z^1 对应的函数 lnor 中隐含 $2n_2$ 个参数。

第6.1节中，矩阵 E 的参数量为 $2n_2$，矩阵 $\hat{W}^1$ 的参数量为 n_2^2，向量 $\hat{\boldsymbol{b}}^1$ 的参数量为 n_2。第6.2节中，矩阵 $\hat{W}^2$ 的参数量为 n_2^2，向量 $\hat{\boldsymbol{b}}^2$ 的参数量为 n_2，向量 $\hat{\boldsymbol{b}}$ 的参数量为 n_1。

将这些数量相加，即得 BERT 模型待定参数的数量

$$n_8(n_2^2 + 3n_2n_6 + 2n_2n_7 + 4n_2 + 3n_6 + n_7) + n_1n_2 + 2n_2^2 + n_2n_3 + 6n_2 + n_1.$$

当 $n_1 \sim n_8$ 的取值为第3节中的典型值时，待定参数的数量为 35945786。

参考文献

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805. 2018.4

[2] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, et al. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. arXiv:1909.11942. 2019.11

[3] Zhilin Yang, Zihang Dai, Yiming Yang, et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding. arXiv:1906.08237. 2019.6

[4] Yinhan Liu, Myle Ott, Naman Goyal, et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692. 2019.7

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention Is All You Need. arXiv:1706.03762. 2017.3

[6] Gage, Philip. A New Algorithm for Data Compression. The C User Journal. 1994

[7] A New Algorithm for Data Compression. Dr. Dobb's Journal. 1 February 1994. Retrieved 10 August 2020